COMPARISON OF FRONT END FEATURES IN HMM BASED DIGIT RECOGNITION

by

ROHIT SINHA

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

December, 1998

COMPARISON OF FRONT END FEATURES IN HMM BASED DIGIT RECOGNITION

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

by
ROHIT SINHA

to the
DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
December, 1998

CERTIFICATE

This is to certify that the work contained in the thesis entitled, "**Comparison of Front-End Features in HMM based Digit Recognition**", has been carried out by **Rohit Sinha** under my supervision, and it has not been submitted elsewhere for a degree.

[illegible]

Dr. S. Umesh
Asst. Professor
Dept. of Electrical Engg.
Indian Institute of Technology
Kanpur

December, 1998

Acknowledgements

I take this opportunity to express my deep gratitude towards my thesis supervisor Dr. S. Umesh for his kind hearted support and encouragement throughout this thesis work. I would like to thank him for providing me the best possible facilities and motivation during occasional moments of despair. I would also like to thank my teachers at IITK for enriching my knowledge through their erudite lectures.

I am grateful to Shafi Ullah Khan and Ashok, for providing many tips regarding PC and various softwares. I also wish to thank my lab mates Sandeep and Anju for their excellent company and occasional help.

Finally, I wish to thank my family and friends for their support. I am especially grateful to my parents for the immeasurable love and support I received from them. Most of all, I thank my wife, Nidhi and son, Sankalp for their patience and understanding not only during this thesis work but throughout the M.Tech. programme.

Rohit Sinha
December 1998

Abstract

In this work, we have evaluated the performance of continuous digit recognizer on two most popularly used front-end features, namely LPC- and MEL-cepstral features so as to study the robustness of the above said features for digit recognition task For this purpose we have implemented a HMM based continuous digit recognizer using development environment provided by CSLU-Toolkit Our study finds that MEL-cepstral feature provides slightly better word and sentence level accuracies compared to LPC-cepstral features Our study also indicates that for a fixed product of number of states and number of mixtures per state in a model, the models with higher number of state result in better word as well as sentence level accuracy In the later part of this work, we have reported a study on the relationship between any two speakers which is important from the point of view of gaining an insight into the development of speaker-independent speech recognition systems

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Speech Recognition has a long history of being one of the difficult problems in Artificial Intelligence and Computer Science Work in speech recognition predates the invention of computers However, the serious work in speech recognition started in the late fifties with the advent of digital computers equipped with A/D converters The problems of segmentation, classification, and pattern matching were explored in the sixties and a small vocabulary connected speech task was demonstrated In the early seventies, the role of syntax and semantics in connected speech recognition was explored and demonstrated The seventies also witnessed the development of a number of basic techniques such as dynamic time warping, network representation, probabilistic modeling, beam search, and forward-backward algorithm The early eighties witnessed a trend towards practical systems with very large vocabularies but computational and accuracy limitation made it necessary to require one-word-at-a-time with a short pause between them In nineties, the emphasis shifted to natural language front ends to the recognizer and task shifted to commercial application specific ones At the same time, speech recognition technology has been increasingly used within telephone networks to automate as well as enhance operator services

Research in speech recognition has followed two primary roots those adopting a knowledge-based approach, and those adopting a statistically data-based approach *Knowledge-based* approach to speech recognition and understanding [11] have attempted to express human knowledge of speech in terms of acoustic-phonetic rules based on specific feature of the acoustic waveform For these approaches, the acoustic signal is first segmented and labelled into phoneme-like units, and then resulting phoneme string is used for lexical and syntactic analysis Words in the lexicon are represented in terms of the phonemic spellings and syntax is usually described by conventional linguistic means It is known that a human can be trained to *read* speech spectra which supports the existence of distinct features in speech spectrum [27], but the machine realization of this human ability is far poorer than the well trained human expert The possible reasons for the poor performance of this approach are the absence of good understanding of the human auditory mechanism and the inability of human experts to formalize completely their knowledge Thus speech recognition remains an unsolved problem for the knowledge-based approach

In contrast to the knowledge-based approach, the *data-based* statistical approaches have achieved considerable success. These are based on modeling the speech signal by itself by some well-defined statistical algorithms that can automatically extract *knowledge* from speech data The predominant class of these algorithms is the hidden Markov model (HMM) [7, 13] An HMM-based system depends on three key factors

1 a detailed modeling scheme which is capable of accommodating various uncertainties,

2 access to sufficient speech data, and

3 an automatic learning algorithm to improve the recognition accuracy

By using HMMs, the speech signal variability in parameter space and time can be modeled effectively Unlike the knowledge-based approach, the HMM learning procedure is achieved by presenting the speech data to HMMs and automatically improving the models by data as opposed to some heuristic rules presented by human experts In general, the more data presented to the model, the higher the recognition accuracy achieved HMM methods have presented speech recognition with a sound theoretical basis, and have resulted in significant advances in large-vocabulary continuous speaker-independent speech recognition [13]

The HMM can be based on either discrete output probability distributions or continuous output probability density functions, which are very important to acoustic modeling Both discrete HMM and continuous HMM are widely used in state-of-the-art speech recognition systems [13, 17, 21] For the discrete HMM, vector quantization (VQ) makes it possible to use a non-parametric, discrete output probability distribution to model the observed speech signals The objective of VQ is to find the set of reproduction vectors, or codebook, that represents an information source with minimum distortion Under discrete HMM framework, VQ is first used to obtain discrete output symbols The discrete HMM then models observed discrete symbols In contrast, the continuous mixture HMM [20] uses continuous mixture probability density functions to model speech parameters directly without using VQ, and usually needs extensive training data and computation times

On the other hand, the semi-continuous HMM [7] which is a very general model including both discrete and continuous mixture HMMs as its special forms, unifies VQ, the discrete HMM, and the continuous mixture HMM

1.1 Scope and Organization of the Thesis

In speech recognition systems, researchers first of all used linear-prediction based features, later the use of Mel-cepstral features reportedly improved the recognition accuracy We have tried to find out in this work how the choice of a particular front-end feature is related to the number of states and number of mixtures per state in the model used in the recognition system Such a study is of importance as it reveals the robustness of a particular feature in HMM based recognition systems

In this thesis, we will provide the background information on the hidden Markov modeling of speech in Chapter 2 In the same chapter we will describe how HMMs are used for speech recognition

In Chapter 3, we will describe the various front-end features used in our study In Chapter 4, we will describe the database used for this study and the recognizer implementation Chapter 5 contains the summary of our experimental results and the conclusion

Finally, in Chapter 6, we will present an additional study about the relationship between speakers which could to provide an important clue in realization of speaker independent speech recognition systems

Chapter 2

Hidden Markov Models

2.1 Definition of HMM

The Hidden Markov Model is a finite set of *states*, each of which is associated with (generally multidimensional) probability distribution [19] Transitions among states are governed by the set of probabilities called *transition probabilities* In a particular state the outcome or observation can be generated, according to the associated probability distribution It is only the outcome, not the state is visible to an external observer and therefore the states are "hidden" to the outside hence the name Hidden Markov Model (HMM)

In order to define an HMM completely, following elements are needed

- The number of states of the model, N
- The number of observation symbols in alphabet, M If the observation is continuous then M is infinite
- A set of state transition probabilities, $\Lambda = \{a_{ij}\}$

  $$a_{ij} = P\{q_{t+1} = j \mid q_t = i\}, \quad 1 \leq i, j \leq N$$

  where q_t denoted the current state

  Transition probabilities should satisfy the normal stochastic constraints,

  $$a_{ij} \geq 0, \quad 1 \leq i, j \leq N$$

and

$$\sum_{j=1}^{N} a_{ij} = 1, \quad 1 \leq i \leq N$$

- A probability distribution in each state, $B = \{b_j(k)\}$,

$$b_j(k) = P\{o_t = v_k \mid q_t = j\}, \quad 1 \leq j \leq N, 1 \leq k \leq M$$

where v_k denotes the k^{th} observation symbol in the alphabet, and o_t the current parameter vector

Following stochastic constraints must be satisfied,

$$b_j(k) \geq 0, \ \ 1 \leq j \leq N, 1 \leq k \leq M$$

and

$$\sum_{k=1}^{M} b_j(k) = 1, \ \ 1 \ \leq j \leq N$$

If the observation are continuous then we will have to use a continuous probability density function, instead of a set of discrete probabilities In this case we specify the parameters of the probability density function Usually the probability density is approximated by a weighted sum of M Gaussian distribution $\aleph$,

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} \, \aleph(\mu_{jm}, \Sigma_{jm}, o_t)$$

where,

c_{jm} = weighting coefficient
μ_{jm} = mean vectors
Σ_{jm} = covariance matrices

c_{jm} should satisfy the stochastic constraints,

$$c_{jm} \geq 0, \quad 1 \leq j \leq N, 1 \leq m \leq M$$

and

$$\sum_{m=1}^{M} c_{jm} = 1, \ 1 \leq j \leq N$$

- The initial state distribution, $\pi = \{\pi_i\}$
  where,

$$\pi_i = P\{q_t = i\}, \quad 1 \leq i \leq N$$

Therefore we can use the compact notation

$$\lambda = (\Lambda, B, \pi)$$

to denote an HMM with discrete probability distributions,while

$$\lambda = (\Lambda, c_{jm}, \mu_{jm}, \Sigma_{jm}, \pi)$$

to denote one with continuous densities

2.2 Assumptions in The Theory of HMMs

For the sake of mathematical and computational tractability following assumptions are made in the theory of HMMs

1 *The Markov Assumption* As given in the definition of HMMs, the transition probabilities are defined as,

$$a_{ij} = P\{q_{t+1} = j \mid q_t = i\}$$

In other words it is assumed that the next state is dependent only upon the present state This is called the Markov assumption and the resulting model becomes actually a first order HMM However generally the next state may be dependent on past k states and it is possible to obtain such model, called k^{th} order HMM But it is seen that the a higher order HMM will have higher complexity Even though first order HMMs are the most common, some attempts have been made to use higher order HMMs also

2 *The Stationary Assumption* Here it is assumed that the state transition probabilities are independent of the actual time at which the transition takes place Mathematically,

$$P\{q_{t_1+1} = j \mid q_{t_1} = i\} = P\{q_{t_2+1} = j \mid q_{t_2} = i\}$$

for any t_1 and t_2

3 *The Output Independence Assumption* This is the assumption that the current output(observation) is statistically independent of the previous outputs (observations) We can formulate this assumption mathematically, by considering a sequence of observations,

$$\mathbf{O} = o_1, o_2, o_3, \ldots, o_T$$

Then according to the assumption for an HMM λ,

$$P\{\mathbf{O} \mid q_1, q_2, \ldots, q_T, \lambda\} = \sum_{t=1}^{T} P(o_t \mid q_t, \lambda)$$

However unlike other two, this assumptions has a very limited validity In some cases this assumption may not be fair enough and becomes a severe weakness of HMMs

2.3 The Three Basic Problems for HMMs

Once we have the HMM, three basic problem of interest must be solved for the model to be useful for the real-world applications

1 *The Evaluation Problem* Given an HMM λ and a sequence of observations $\mathbf{O} = o_1, o_2, \ldots, o_T$, what is the probability that the given observation sequence is generated by the model, $P\{\mathbf{O} \mid \lambda\}$?

2 *The Decoding Problem* Given a model λ and the sequence of observations $\mathbf{O} = o_1, o_2, \ldots, o_T$, what is the most likely state sequence in the model that produced the observations?

3 *The Learning Problem* Given the model λ and a sequence of observations, how should the parameter of the model (Λ, B, π) be adjusted in order to maximize $P\{\mathbf{O} \mid \lambda\}$?

Evaluation problem can be used for the isolated (word) recognition Decoding problem is related to the continuous recognition as well as the segmentation Learning problem must be solved, if we want to train an HMM for the subsequent use of recognition tasks

2.3.1 The Evaluation Problem and Forward Algorithm

We have a model $\lambda = (\Lambda, B, \pi)$ and a sequence of observations $\mathbf{O} = o_1, \quad , o_T$ and $P(\mathbf{O} \mid \lambda)$ must be found We can compute this quantity using simple probabilistic arguments But this computation involves number of operations of the order of N^T This is very large even if the length of the sequence, T, is moderate Therefore we use another method for this computation which has a considerably low complexity and makes use an auxiliary variable $\alpha_t(i)$, called the *forward variable* The forward variable is defined as the probability of the partial observation sequence $o_1, o_2, \quad , o_t$, when it terminates at the state i Mathematically,

$$\alpha_t(i) = P\{o_1, o_2, \quad , o_t, q_t = i \mid \lambda\} \tag{2 1}$$

then it is easy to see that following recursive relationship holds,

$$\alpha_{t+1}(j) = b_j(o_{t+1}) \sum_{i=1}^{N} \alpha_t(i) a_{ij}, \quad 1 \le j \le N, \; 1 \le t \le T-1 \tag{2 2}$$

where,

$$\alpha_1(j) = \pi_j b_j(o_1), \quad 1 \le j \le N$$

Using this recursion we can compute, $\alpha_T(i), \; 1 \le i \le N$ and then the required probability is given by,

$$P\{\mathbf{O} \mid \lambda\} = \sum_{i=1}^{N} \alpha_T(i) \tag{2 3}$$

The complexity of this method, known as *forward algorithm* is proportional to N^2T, which is linear with respect to T whereas direct computation mentioned earlier, had an exponential complexity

In a similar way we can define the *backward variable* as the probability of the partial observation sequence $o_{t+1}, o_{t+2}, \quad , o_T$, given that the current state is i Mathematically,

$$\beta_t(i) = P\{o_{t+1}, o_{t+2}, \quad , o_T, q_t = i \mid \lambda\} \qquad (2\ 4)$$

As in the case of $\alpha_t(i)$ there is a recursive relationship which can be used to compute $\beta_t(i)$ efficiently,

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(o_{t+1}), \quad 1 \leq i \leq N, 1 \leq t \leq T-1 \qquad (2\ 5)$$

where,

$$\beta_T(i) = 1, \quad 1 \leq i \leq N$$

Further we can see that,

$$\alpha_t(i)\beta_t(j) = P\{\mathbf{O}, q_t = i \mid \lambda\}, \quad 1 \leq i \leq N, \ 1 \leq t \leq T \qquad (2\ 6)$$

Therefore this gives another way to compute $p\{\mathbf{O} \mid \lambda\}$, by using both forward and backward variables as given below,

$$P\{\mathbf{O} \mid \lambda\} = \sum_{i=1}^{N} P\{\mathbf{O}, q_t = i \mid \lambda\} = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \qquad (2\ 7)$$

2.3.2 The Decoding Problem and Viterbi Algorithm

In this case we want find the most likely state sequence for a given sequence of observations $\mathbf{O} = o_1, o_2, \quad , o_T$ and the model $\lambda = (\Lambda, B, \pi)$ The solution to this problem depends upon the way "most likely state sequence" is defined One approach is to find the most likely state q_t at time $t = t$ and to concatenate all such $q_t's$ But some time this method does not give physically meaningful state sequence Therefore we would go for another method which has no such problem In this method, commonly known as *Viterbi algorithm* [5, 24], the whole state sequence with the maximum likelihood is found In order to facilitate the computations we define an auxiliary variable,

$$\delta_t(i) = \max_{q_1 q_2 \quad q_{t1}} P\{q_1, q_2, \quad , q_{t-1}, q_t = i, o_1, o_2, \quad , o_{t-1} \mid \lambda\}, \tag{2 8}$$

which gives the highest probability that partial observation sequence and state sequence can have,when the current state is i It is easy to observe that the following recursive relationship holds

$$\delta_{t+1}(j) = b_j(o_{t+1}) \left[\max_{1 \le i \le N} \delta_t(i) a_{ij} \right], \ 1 \le i \le N, 1 \le t \le T-1 \tag{2 9}$$

where,

$$\delta_1(i) = \pi_j b_j(o_1), \ \ 1 \le i \le N$$

So the procedure to find the most likely state sequence starts from the calculation of $\delta_T(j)$, $1 \le j \le N$ using the above recursion,while always keeping a pointer to the "winning state" in the maximum finding operation Finally state j^*, is found where

$$j^* = \arg \max_{1 \le j \le N} \delta_T(j), \tag{2 10}$$

and starting from this state, the sequence of the states is back traced as the pointer in each state indices This gives the required set of states This whole algorithm can be interpreted as a search in a graph whose nodes are formed by the states of the HMM in each of the time instants t, $1 \le t \le T$

2.3.3 The Learning Problem

Generally, the learning problem is to adjust the HMM parameters, so that the given set of observations (called the *training set*) are represented by the model in best way for the intended application Thus it would be clear that the "quantity" we wish to optimize will be different from application to application In other there may be several *optimization criteria* for learning, out of which suitable one is selected depending on the application

There are two main optimization criteria found in ASR literature, Maximum Likelihood(ML) and Maximum Mutual Information(MMI)[15] Since we have used ML optimization criterion in our experiments so the solution to the learning problem under this criterion is described in detail the following sections, while only basic principle is discussed for MMI criterion

Maximum Likelihood(ML) Criterion

In ML criterion we try to maximize the probability of a given sequence of observation $\mathbf{O}^w$, belonging to a class w, given the HMM λ_w of the class w, with respect to the parameters of the model λ_w This probability is the total likelihood of the observations and can be expressed mathematically as,

$$P_{tot} = P\{\mathbf{O}^w \mid \lambda_w\} \qquad (2\ 11)$$

However we consider one class only at a time we can drop super and subscript 'w' Then the ML criterion can be written as

$$P_{tot} = P\{\mathbf{O} \mid \lambda\} \qquad (2\ 12)$$

However there is no known way to solve this analytically for the model $\lambda = (\Lambda, B, \pi)$, which maximizes the quantity P_{tot} But we can choose model parameters such that it is locally maximized, using a iterative procedure, like Baum-Welch method or a gradient based method

Baum-Welch Algorithm This method can be derived using simple "occurrence counting" arguments or using calculus to maximize the auxiliary quantity called Q-function, defined as,

$$Q(\lambda, \overline{\lambda}) = \sum_q P(\mathbf{O}, q \mid \lambda) \log P(\mathbf{O}, q \mid \overline{\lambda}) \qquad (2.13)$$

over $\overline{\lambda}$ Because

$$Q(\lambda, \overline{\lambda}) \geq Q(\lambda, \lambda) \Rightarrow P(\mathbf{O} \mid \overline{\lambda}) \geq P(\mathbf{O} \mid \lambda) \tag{2 14}$$

we can maximize the function $Q(\lambda, \overline{\lambda})$ over λ to improve $\overline{\lambda}$ in the sense of increasing the likelihood $P(\mathbf{O} \mid \overline{\lambda})$ Eventually the likelihood function converges to a critical point if we iterate the procedure [2]

To describe the Baum-Welch algorithm, (also known as *Forward-Backward algorithm*), we need to define two more variables, in addition to the forward and backward variables defined in previous section These variables can however be expressed in terms of forward and backward variables

First one of those variables is defined as probability of being in state i at $t = t$ and in state j at $t = t + 1$ Formally

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j \mid \mathbf{O}, \lambda) \tag{2 15}$$

This is same as,

$$\xi_t(i, j) = \frac{P(q_t = i, q_{t+1} = j, \mathbf{O} \mid \lambda)}{P(\mathbf{O} \mid \lambda)} \tag{2 16}$$

Using the forward and backward variables this can be expressed as,

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)} \tag{2 17}$$

The second variable is a-posteriori probability,

$$\gamma_t(i) = P(q_t = i \mid \mathbf{O}, \lambda) \tag{2 18}$$

that is the probability of being in the state i at $t = t$, given the observation sequence and the model In forward and backward variables this can be expressed as,

$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)} \tag{2 19}$$

One can see that the relation between $\gamma_t(i)$ and $\xi_t(i, j)$ is given by,

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j), \quad 1 \leq i \leq N, \ 1 \leq t \leq M \tag{2 20}$$

Now it is possible to describe the Baum-Welch learning process, where the parameters of HMM is updated in such a way to maximize the quantity $P(\mathbf{O} \mid \lambda)$ Assuming the starting model $\lambda = (\Lambda, B, \pi)$, we calculate the $\alpha's$ and $\beta's$ using the recursions (2 2) and (2 5), and then $\xi's$ and $\gamma's$ using (2 17) and (2 20) Next step is to update the HMM parameter according to Eq (2 21) to (2 23), known as *re-estimation formulas*

$$\overline{\pi}_i = \gamma_1(i), \quad 1 \leq i \leq N \tag{2 21}$$

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \quad 1 \leq i,j \leq N \tag{2 22}$$

$$\overline{b}_j(k) = \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}, \quad 1 \leq i,j \leq N \tag{2 23}$$

These re-estimation formulas can be easily modified to deal with the continuous density case too and the re-estimation formulas for the coefficient of the mixture density, i e , c_{jk}, μ_{jk}, and Σ_{jk}, are of the form [19],

$$\overline{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{2 24}$$

$$\overline{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)\ o_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2 25}$$

$$\overline{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)\ (o_t - \mu_{jk})(o_t - \mu_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2 26}$$

where prime denotes vector transpose and where $\gamma_t(j,k)$ is the probability of being in the state j at time t with the k^{th} mixture accounting for o_t, i e ,

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_T(j)\beta_t(j)}\right] \left[\frac{c_{jk}\aleph(o_t, \mu_{jk}, \Sigma_{jk})}{\sum_{m=1}^{M} c_{jm}\aleph(o_t, \mu_{jm}, \Sigma_{jm})}\right] \tag{2 27}$$

Gradient Based Method In gradient based method, any parameter Θ of the HMM λ is updated according to standard formula,

$$\Theta^{new} = \Theta^{old} - \eta \left[\frac{\partial J}{\partial \Theta} \right]_{\Theta = \Theta^{old}} \tag{2 28}$$

where J is the quantity to be minimized We define in this case,

$$J = E_{ML} = -\log(P\{\mathbf{O} \mid \lambda\}) = -\log(P_{tot}) \tag{2 29}$$

Since minimization of $J = E_{ML}$ is equivalent to the maximization of P_{tot}, Eq (2 28) yield the required ML optimization criterion The derivative $\frac{\partial J}{\partial \Theta}$ for any parameter Θ of the model can be obtained by relating J to the model parameters via L_{tot} As a key step to do so, using Eqs (2 7) and (2 12) we can obtain,

$$P_{tot} = \sum_{i=1}^{N} P\{\mathbf{O}, q_t = i \mid \lambda\} = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \tag{2 30}$$

Differentiating the last quantity in Eq (2 29) with respect to an arbitrary parameter Θ,

$$\frac{\partial J}{\partial \Theta} = -\frac{1}{P_{tot}} \frac{\partial P_{tot}}{\partial \Theta} \tag{2 31}$$

Eq (2 31) gives $\frac{\partial J}{\partial \Theta}$, if we know $\frac{\partial P_{tot}}{\partial \Theta}$ which can be found using Eq (2 30) Since there are two main parameter sets in HMM, namely transition probabilities a_{ij}, $1 \leq i, j \leq N$ and observation probabilities $b_j(k)$, $1 \leq j \leq N, 1 \leq k \leq M$, we can find the derivative $\frac{\partial P_{tot}}{\partial \Theta}$ for each of the parameter sets and hence the gradient $\frac{\partial J}{\partial \Theta}$

To find the gradient with respect to transition probabilities using the chain rule,

$$\frac{\partial P_{tot}}{\partial a_{ij}} = \sum_{t=1}^{T} \frac{\partial P_{tot}}{\partial \alpha_t(j)} \frac{\alpha_t(j)}{\partial a_{ij}} \tag{2 32}$$

By differentiating Eq (2 30) with respect to $\alpha_t(j)$ we get,

$$\frac{\partial P_{tot}}{\partial \alpha_t(j)} = \beta_t(j) \tag{2 33}$$

and differentiating a time shifted version of Eq (2 2) with respect to a_{ij},

$$\frac{\partial \alpha_t(j)}{\partial a_{ij}} = b_j(o_t)\alpha_{t-1}(i) \tag{2 34}$$

Eqs (2 32),(2 33)and(2 34) give, $\frac{\partial P_{tot}}{\partial a_{ij}}$, and substituting this quantity in Eq (2 31), we get the required result

$$\frac{\partial J}{\partial a_{ij}} = -\frac{1}{P_{tot}} \sum_{t=1}^{T} \beta_t(j) b_j(o_t) \alpha_{t-1}(i) \tag{2 35}$$

Similarly to find the gradient with respect to observation probabilities again using the chain rule,

$$\frac{\partial P_{tot}}{\partial b_j(o_t)} = \frac{\partial P_{tot}}{\partial \alpha_t(j)} \frac{\alpha_t(j)}{\partial b_j(o_t)} \tag{2 36}$$

Differentiating a time shifted version of Eq (2 2) with respect to $b_j(o_t)$,

$$\frac{\partial \alpha_t(j)}{\partial b_j(o_t)} = \frac{\alpha_t(j)}{b_j(o_t)} \tag{2 37}$$

Finally we get the required probability, by substituting $\frac{\partial P_{tot}}{\partial b_j(o_t)}$ in Eq (2 31), which is obtained by substituting Eqs (2 37) and (2 33) in Eq (2 36)

$$\frac{\partial J}{\partial b_j(o_t)} = -\frac{1}{P_{tot}} \frac{\alpha_t(j)\beta_t(j)}{b_j(o_t)} \tag{2 38}$$

Usually this equation is given the following form, by first substituting for P_{tot} from Eq (2 30) and then substituting from Eq (2 20)

$$\frac{\partial J}{\partial b_j(o_t)} = -\frac{\gamma_t(j)}{b_j(o_t)} \tag{2 39}$$

If the continuous densities are used then $\frac{\partial J}{\partial c_{jk}}$, $\frac{\partial J}{\partial \mu_{jk}}$ and $\frac{\partial J}{\partial \Sigma_{jk}}$ can be found by further propagating the derivative $\frac{\partial J}{\partial b_j(o_t)}$ using the chain rule

Maximum Mutual Information(MMI) Criterion

In ML criterion we optimize an HMM for only one class at a time, and do not touch the HMMs for other class at that time This procedure does not involve the concept of "discrimination" which plays an important role in Pattern Recognition and of course in ASR Thus ML learning procedure gives a poor discrimination ability to HMM system, specially when the estimated parameters (in training phase) do not match with speech inputs used in recognition phase This type of mismatch can arise due to two reasons One the training and recognition data have considerably different statistical properties, and the other is the difficulties in obtaining reliable parameters estimates in the training The MMI criterion on the other hand considers HMMs of all classes simultaneously, during the training Parameters of the correct model are updated to enhance it's contribution to the observations, while parameters of the alternative models are updated to reduce their contributions This procedure provides a high discriminative ability to the HMM system and thus MMI belongs to so called "discriminative training" category

2.4 Use of HMMs in Isolated Recognition

Isolated recognition in general is based on any kind of isolated speech unit, which can be word, a sub-word or even a concatenation of words However only isolated word recognition has direct practical applications, while the isolated recognition based on other two types of units has basically theoretical value Specially the sub-word recognition in isolated mode can give a good indication about the continuous recognition based on the same techniques

In a simple isolated speech unit recognition, where the vocabulary contains V speech units, we can use the system depicted in Figure 2 1 There are however many possible ways to provide solution of the task, because there are many optimality criteria, and several algorithmic implementation of a particular criterion Out of those, only Baum-Welch method, as the one used ın ımplementatıon, ıs descrıbed below Our task ıs comprısed of two sub tasks, namely the traınıng and the recognıtıon

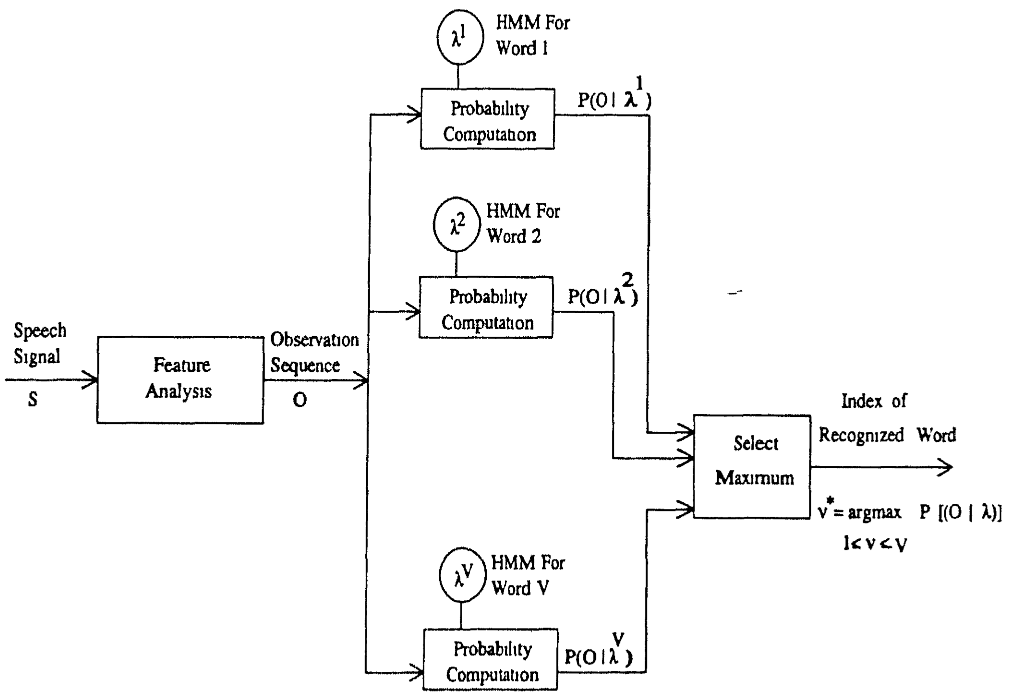


Fıgure 2 1 Block dıagram of an ısolated word HMM recognızer (after Rabıner [20])

2.4.1 Training

The traınıng procedure can be summarızed as follows:

1 Inıtıalıze each HMM, $\lambda_i = (A_i, B_i, \pi_i)$, $1 \leq I \leq N$ wıth values generated randomly or usıng an ınıtıalızatıon algorithm lıke *segmental k-means* [8]

2 Take an observation sequence and

   - calculate the forward and backward probabilities for each HMM, using Eqs (2 2) and (2 5)
   - using Eq (2 17) and (2 20), calculate the auxiliary variables ξ and γ
   - update parameters in each model, using Eqs (2 21), (2 22) and (2 23)

3 Go to step 2, unless all observation sequences are considered

4 Repeat step 2 to 3 until a convergence criteria is satisfied

This procedure can be easily modified for the continuous density case, by using the Eqs (2 24), (2 25) and (2 26) to reestimate the coefficients of the mixture density instead of Eq (2 23) in step 4 of above training procedure

2.4.2 Recognition

Comparative to the training the recognition is much simpler and the procedure is given below

1 Take the observation sequence to be recognized and

   - calculate the forward and backward probabilities for each of HMM using the recursion (2 2) and (2 5)
   - using Eq (2 12) calculate the likelihood,

$$P_m^w, \quad 1 \leq m \leq M$$

2 The recognized class w^*, to which the observation sequence belongs, is given by

$$w^* = arg \max_{1 \leq m \leq M} P_m^w.$$

3 Go to step 2, unless all observation sequence to be recognized are considered

The recognition rate in this case can be calculated as the ratio of the correctly recognized speech units and the total number of speech units (in the observation sequence) to be recognized

2.5 Use of HMM in Continuous Recognition

In isolated recognition we have used one HMM for each speech unit But in continuous recognition, this is not possible because a continuous sequence of speech units, which is usually called a *sentence*, is to be recognized and hence the number of possible sentences may be quite high even for a small vocabulary In addition there are other two fundamental problems associated with continuous recognition

1 We do not know the end points of the speech units contained in the sentence

2 We do not know how many speech units are contained in the sentence

Because of the problems mentions above, the continuous recognition is more complicated than isolated recognition However HMMs a provide good frame work for continuous recognition also. In this case we connect the HMMs for each speech units in a sentence to form a large HMM in a manner shown in Figure 2 2 This representation is called *clamped* grammar case HMM in the *free* grammar case is obtained by connecting all speech units in the vocabulary as shown in Figure 2 3 The transitions between the speech units are derived using so called *language model*

A block diagram of a canonic system for connected digit recognition is shown in Figure 2 4 There are three basic steps in the recognition process, namely·

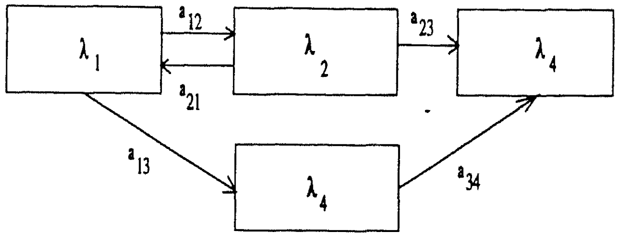


Figure 2 2 Sentence HMM in clamped grammar case shown with 4 speech units

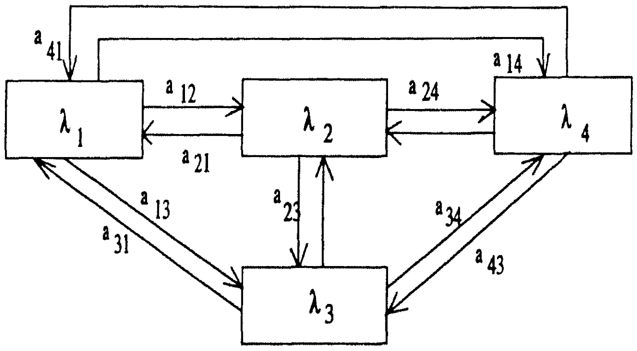


Figure 2 3 Sentence HMM in free grammar case shown with 4 speech units

1 *Spectral analysis*, in which the speech signal, $s(n)$, is converted to an appropriate spectral representation, e g , filter-bank vector, LPC-based vector

2 *Connected word pattern matching*, in which the sequence of spectral vectors of the unknown (test) connected digit string is matched against whole word (single digit) patterns using any of the algorithms, viz , Level building algorithm, Two-level DP algorithm etc The output of this process is a set of candidate digit strings, generally of different lengths, ordered by distance (likelihood, probability) score

3. *Postprocessing*, in which the candidate digit strings are subjected to further processing (based on digit duration, word stress, etc) so as

to elıminate unreasonable (unlıkely) candıdates The postprocessor chooses the most lıkely dıgıt strıng from the ordered lıst of candıdates whıch passed the postprocessor tests

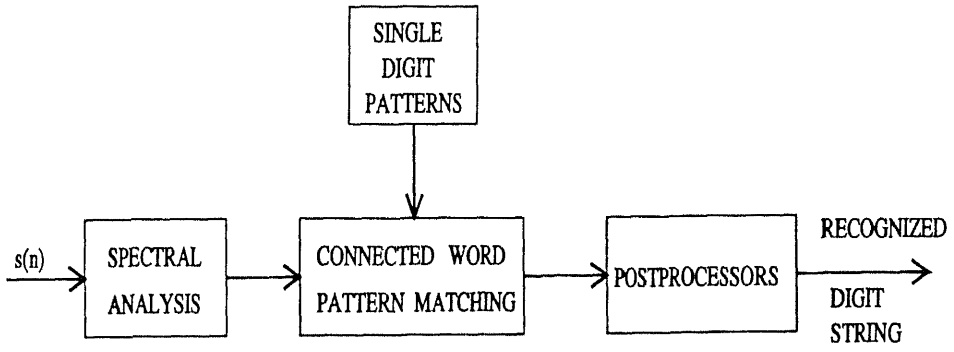


Fıgure 2 4 Block dıagram of connected dıgıt recognıtıon process

2.5.1 Statistical Language Model

The goal of language model, as already mentıoned, ıs to estımate the probabılıty of a sequence of speech unıts Assume that **w** ıs a specified sequence of L speech unıts,

$$\mathbf{w} = w_1, w_2, \quad , w_L$$

were ınterested ın the probabılity, $P\{\mathbf{w}\}$ Thıs can be approxımated by,

$$P_N\{\mathbf{w}\} = \Pi_{i=1}^{L} p\{w_i \mid w_{i-1}, w_{i-2}, \cdot \ , w_{i-N+1}\}$$

whıch ıs called an *N-gram language model* These probabılitıes can be evaluated by occurrence countıng ın the database However ıt ıs clearly ımpractıcal except for the case of N=2 or at the most N=3 In the case of N=2, we call ıt *bıgram language model* and ıt ıs the most wıdely used case [19] An extreme case of bıgram model called *word paır model* can be obtaın by quantızıng the bıgram probabılıtıes between 0 and 1 But the most relaxed grammar ıs *no grammar* where any speech unıt can be followed every speech unıt ın the vocabulary

2.5.2 Training

The procedure of traınıng the model usıng the gradıent based ML crıterıon ıs summarızed as below

1 Inıtıalıze each HMM $\lambda_i = (\Lambda_i, B_i, \pi_i)$, $1 \le i \le N$ wıth values generated randomly or usıng an ınıtıalızatıon algorıthm lıke *segmental k-means*

2 Take the observatıon sequence of a sentence and

- Form the correspondıng sentence model using the HMMs of the speech unıts contaıned ın the sentence
- Calculate the forward and backward probabılıtıes usıng the recursıon (2 2) and (2 5)
- Usıng the Eq (2 30) calculate the lıkelıhood of the observatıons ın the sentence model
- Usıng Eqs (2 35) and (2 38) calculate the gradıent wıth respect to all parameters in sentence model
- Update all parameters ın each HMMs of the sentence model usıng Eq (2 28)

3 Go to step 2, unless all observatıon sequences are consıdered

4 Repeat step 2 and 3, untıl a convergence crıterıon ıs satısfied

2.5.3 Recognition

In the recognıtıon phase ıt ıs assumed that traıned HMMs for each of the speech unıts ın the vocabulary is avaılable The task ıs to find the underlyıng speech unıt sequence, gıven an observatıon sequence O correspondıng to an unknown sentence Mathematıcally thıs operatıon can be expressed as,

$$\mathbf{w}^* = arg \max_{\mathbf{w}} P\{\mathbf{w}, \mathbf{O} \mid \Lambda\} \tag{2 40}$$

where $\mathbf{w} = w_1, w_2, \ldots, w_S$ is an arbitrary speech unit sequence of arbitrary length S Since $P\{\mathbf{w}\}$ is provided by the statistical language model, so to find $\mathbf{w}^*$ we have to only calculate $P\{\mathbf{O} \mid \mathbf{w}, \Lambda\}$ for every possible $\mathbf{w}$ It is obvious that this procedure will be computationally very expensive, so a cheaper solution is to approximate the procedure, by finding the most likely state sequence, $\mathbf{q}^*$ in the language model Λ, instead of speech unit sequence $\mathbf{w}^*$ Formally

$$\mathbf{q}^* = arg \max_{\mathbf{q}} P\{\mathbf{q}, \mathbf{O} \mid \Lambda\} \tag{2 41}$$

Then it is possible to trace for the corresponding speech unit sequence, via the state sequence In order to calculate $\mathbf{q}^*$ we can use the Viterbi algorithm directly, or a method called *Level building* [12, 21], a variant of Viterbi algorithm Since Viterbi based recognition is suboptimal, unless each speech unit is corresponding to a HMM state, some attempts have been made to develop the efficient method to find the sentence likelihood The so called *N-best algorithm* is one of this

Viterbi-based Search

The Viterbi score, $\delta_t(i)$ can be computed for all the states in the language model Λ at time $t = t$ and then can advanced to the time instant $t = t + 1$, in an inductive manner, as formulated in Eq (2 9) This procedure is known as *time synchronous Viterbi search* because it completely processes at time t before going into the time $t = t + 1$ Finally a backtracking pass gives the required state sequence However even Viterbi search can be very cumbersome if the number of states is large In such cases instead of keeping all candidates at every time instant t, only first T candidates with highest scores are retained This method of pruning search space is known as *beam search*

Level Building

In level building, unlike in Viterbi search, HMM for each speech unit is considered separately The search is performed at various *levels* for each of HMM using Viterbi algorithm, where level corresponds to the position of speech unit within the possible sentence After searching at each level, we find the maximum score for all speech unit models for every time frame t The search in the next level starts with the winning score of the previous level at previous time frame After having completed the searches in l levels, the sequence of winning speech unit model at levels $1, 2, \quad , l$ in that order represents the possible sentence which is l speech units long In order to find the optimum sentence a maximization is done over l

N-best Search

N-best search algorithm is very similar to the time synchronous Viterbi search. Since the purpose of N-best search is to find the optimum speech unit sequence instead of optimum state sequence, a summing operation should be done instead of maximum finding operation However if we completely drop the maximum finding operation it will become forward algorithm and we go back again to the start Therefore the pruning is performed at each state, in addition to the pruning of the beam, keeping only the first N paths with the highest scores This algorithm also does not give theoretically optimum sentence. At the end the algorithm gives N most likely sentences, and for simple task N=1 is enough.

Computing Error Rate

Since we do not know how many speech units are there in the sentence to be recognized, the result of the recognition can have less or more speech units than the correct sentence Therefore the recognized sentence is matched with the correct sentence with help of a dynamic programming algorithm In continuous recognition case, there are three types of errors substitution, deletion, and insertion After finding the percentage word correctly recognized, the insertions, deletions, and substitutions The word accuracy is computed from these as follows

$$Word\ Accuracy = \%\ Word\ correctly\ recognised - \%\ Insertions$$

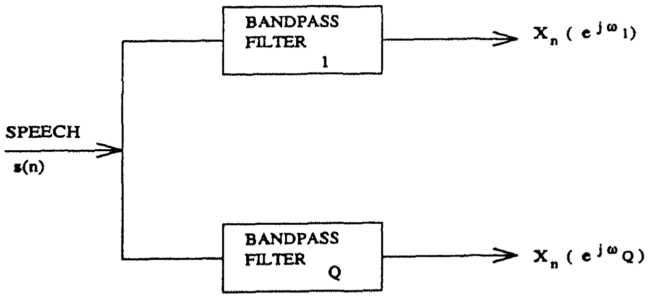
Chapter 3

Front-End Features

The various speech recognizer developed till date can be classified into different classes depending on the varying emphasis to different algorithmic inputs to this field from a wide variety of disciplines, including statistical pattern recognition, communication theory, signal processing, combinatorial mathematics, and linguistics, among others But the most common denominator of all recognition systems is the signal-processing front-end, which converts the speech waveform to some type of parametric representation (generally at a considerably lower information rate) for further analysis and processing

A wide range of techniques exists for parametrically representing the speech signal These include the short time energy, zero crossing rates, level crossing rates, and other related parameters Probably the most important parametric representation of speech is the short time spectral envelope Spectral analysis methods are therefore generally considered as the core of the signal-processing front-end in a speech recognition system There are two dominant methods of spectral analysis-namely, the bank-of-filter spectrum analysis model, and the linear predictive coding (LPC) spectral analysis model

The overall structure of the bank-of-filter model is shown in Figure 3 1 The speech signal, $s(n)$, is passed through a bank of Q bandpass filters whose coverage spans the frequency range of interest in the signal The individual filters can and generally do overlap in frequency domain, as shown at bottom of Figure 3 1 The output of the i^{th} bandpass filter,$X_n(\exp^{j\omega_i})$ (where ω_i is the normalized frequency $2\pi f_i/F_s$, with F_s the sampling frequency) is the short-time spectral representation of the signal $s(n)$, at time n, as seen through the i^{th} bandpass filter with center frequency ω_i It can be easily seen that in the bank-of-filter model each bandpass filter processes the speech signal independently to produce the spectral representation X_n



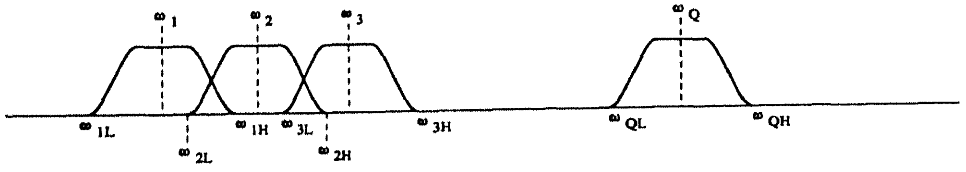


Figure 3 1 Bank-of-filters analysis model

The LPC analysis approach, as illustrated in Figure 3 2, performs spectral analysis on blocks of speech (speech frames) with an all-pole modeling constraint This means that the resulting spectral representation $X_n(\exp^{j\omega})$ is constrained to be of the form $\sigma/A(\exp^{j\omega})$, where $A(\exp^{j\omega})$ is a p^{th} order polynomial with z-transform

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdot + a_p z^{-p}$$

The order, p, ıs called the LPC analysıs order Thus the output of the LPC spectral analysıs block ıs a vector of coefficıents (LPC parameters) that specıfy (parametrıcally) the spectrum of the all-pole model that best matches the sıgnal spectrum over the perıod of tıme ın whıch the frame of speech samples was accumulated

In the next sectıons we wıll be descrıbıng ın brıef both of these sıgnal-processıng front-ends feature processors The mathematıcal detaıls and derıvatıons wıll be omıtted here, for more comprehensıve descrıptıon the ınterested reader ıs referred to [19]

3.1 LPC Front-End Feature Processor

The theory of lınear predıctıon, as applıed to speech, has been well understood for many years [16]. The LPC front end feature has been wıdely used ın speech recognıtıon systems Fıgure 3 3 shows a block dıagram of the LPC feature processor. The basic steps requıred ın thıs feature processıng include the followıng

1 *Preemphasıs.* The dıgıtızed speech sıgnal $s(n)$, ıs put through a low-order dıgıtal system, to spectrally flatten the sıgnal and to make ıt less susceptıble to finite precısıon effects later ın the sıgnal processıng The dıgital system used ın the preemphasızer ıs eıther fixed or slowly

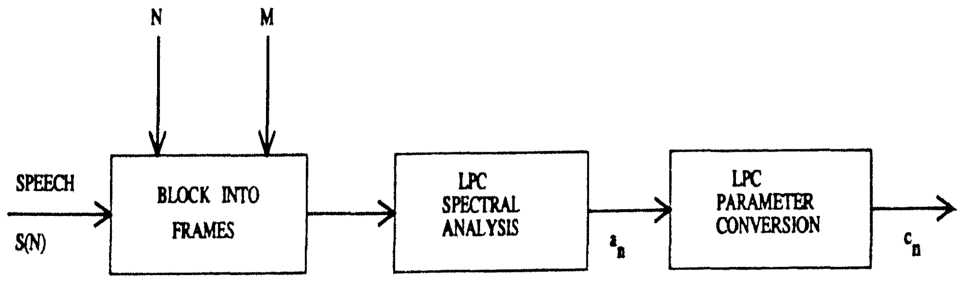


Figure 3 2. LPC analysis model

adaptive (e g , to average transmission conditions, noise backgrounds or even to average signal spectrum) The most widely used preemphasis network is the fixed first-order system,

$$H(z) = 1 - \tilde{a}z^{-1}, \quad 0\,9 \leq \tilde{a} \leq 1\,0$$

2 *Frame Blocking* In this step the preemphasised speech signal, $\tilde{s}(n)$, is blocked into frames of N samples, with adjacent frames being separated by M samples When $M \leq N$, then the adjacent frames will overlap, and the resulting LPC spectral estimates will be correlated from frame to frame, if $M \ll N$, then LPC spectrum estimates will be quite smooth On the other hand, if $M > N$, there will be no overlap between adjacent frames, in fact some of the speech will be totally lost and the correlation between the resulting LPC spectral estimates of adjacent frames will contain a noisy component whose magnitude will increases as M increases This situation is intolerable in any practical LPC analysis for speech recognition If we denote the l^{th} frame of speech by $x_l(n)$, and there are L frames within the entire speech signal, then

$$x_l(n) = \tilde{s}(Ml + n), \quad n = 0, 1, \quad , N-1, \quad l = 0, 1, \quad , L-1$$

3 *Windowing.* The next step in the processing is to window each individual frame so as to minimize the signal discontinuities at the beginning and the end of each frame by tapering the signal to zero at the beginning and end of each frame If we define the window as $w(n)$, $0 \leq n \leq N-1$,

then the result of windowing is the signal

$$\tilde{x}_l(n) = x_l(n)w(n), \quad 0 \leq n \leq N-1$$

A typical window used for the autocorrelation method of LPC is the Hamming window, which has the form

$$w(n) = 0\,54 - 0\,46\cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \leq n \leq N-1$$

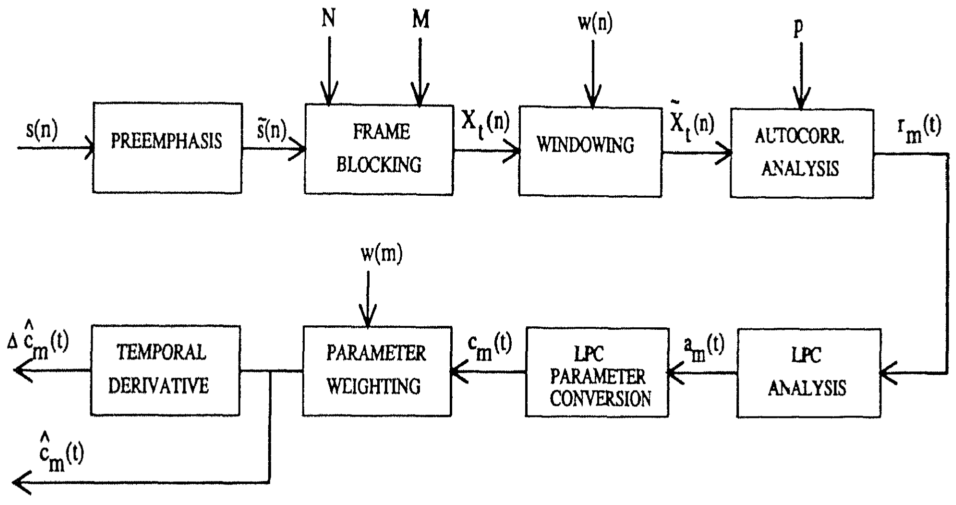


Figure 3 3 Block diagram of LPC processor for speech recognition

4 *Autocorrelation Analysis* Each frame of windowed signal is then autocorrelated to give

$$r_l(m) = \sum_{m=0}^{N-1-m} \tilde{x}_l(n)\tilde{x}_l(n+m), \quad m = 0, 1, \quad , p$$

where the highest autocorrelation value, p, is the order of the LPC analysis. Typically values of p from 8 to 16 have been used The side benefit of the autocorrelation analysis is that the zeroth autocorrelation, $r_l(0)$, is the energy of the l^{th} frame The frame energy is the important parameter for the speech-detection systems

5. *LPC Analysis* The next processing step is the LPC analysis which converts each frame of $p+1$ autocorrelations into an "LPC parameter set", in which the set might be the LPC coefficients, the reflection (or PARCOR) coefficients, the log area ratio coefficients, the cepstral coefficients, or any other desired transformation of the above sets. The formal method for converting from autocorrelation coefficients to an LPC parameter set is known as Durbin's method and can be formally

given as the following algorithm (here the subscript l on $r_l(m)$ is omitted for convenience)

$$
\begin{aligned}
&E^{(0)}=r(0)\\
&k_i = \{r(i) - \textstyle\sum_{j=1}^{L-1} \alpha_j^{(i-1)} r(|i-j|)\}/E^{(i-1)}, \quad 1 \le i \le p\\
&\alpha_i^{(i)} = k_i\\
&\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i\, \alpha_{i-j}^{i-1}\\
&E^{(i)} = (1-k_i^2)E^{(i-1)},
\end{aligned}
$$

The above set of equations are solved recursively for $i = 1, 2, \quad, p$, and the final solution is given as

$$
\begin{aligned}
a_m &= \text{LPC coefficients}=\alpha_m^{(p)}, \quad 1 \le m \le p\\
k_m &= \text{PARCOR coefficients}\\
g_m &= \text{log area ratio coefficients}=\log\left(\tfrac{1-k_m}{1+k_m}\right)
\end{aligned}
$$

6. *Conversion to Cepstral Coefficients* A very important LPC parameter set, which can be derived from the LPC coefficient set, is the LPC cepstral coefficients, $c(m)$ The recursion used is

$$
\begin{aligned}
c_0 &= \ln \sigma^2\\
c_m &= a_m + \textstyle\sum_{k=1}^{m-1} (\frac{k}{m}) c_k a_{m-k}, \quad 1 \le m \le p\\
c_m &= \textstyle\sum_{k=1}^{m-1} (\frac{k}{m}) c_k a_{m-k}, \qquad m > p
\end{aligned}
$$

where σ^2 is the gain term in the LPC model The cepstral coefficients, which are the coefficients of the Fourier transform representation of the log magnitude spectrum, have shown to be a more robust, reliable feature set for speech recognition than the LPC coefficients and its other variants

7 *Parameter Weighting* Since the low-order cepstral coefficients are sensitive to overall slope and the higher-order cepstral coefficients are sensitive to noise, it has become a standard technique to weight the cepstral

coefficients by a tapered window so as to minimize these sensitivities [9]

8 *Temporal Cepstral Derivative* The cepstral representation of the speech spectrum provides a good representation of the local spectral properties of the signal for the given analysis frame An improved representation can be obtained by extending the analysis to include information about the temporal cepstral derivative [6] The time derivative of the log magnitude spectrum has a Fourier series representation of the form

$$\frac{\partial}{\partial t}[\log |S(\exp^{j\omega}, t)|] = \sum_{m=-\infty}^{\infty} \frac{\partial c_m(t)}{\partial t} \exp^{-j\omega m}$$

It is well known that since $c_m(t)$ is a discrete time representation (where t is the frame index), simply using a first-or second-order difference is inappropriate to approximate the derivative as its very noisy Hence a reasonable compromise is to approximate $\partial c_m(t)/\partial t$ by an orthogonal polynomial fit over a finite length window, that is,

$$\frac{\partial c_m(t)}{\partial t} = \Delta c_m(t) \approx \mu \sum_{k=-K}^{K} k c_m(t+k)$$

where μ is an appropriate normalization constant and $(2K+1)$ is the number of frames over which the computation is performed

3.2 Bank-of-Filter Front-End Processor

The bank-of-filter front-end processors can be broadly classified into two categories depending upon the type of filter bank used There are mainly two types of filter bank used for speech recognition, namely the uniform filter bank and non-uniform filter bank.

In uniform filter bank, all filter covering the frequency range of speech, are uniformally spaced. On the other hand the non-uniform filter bank is designed according to some criterion for how the individual filters should be spaced in frequency One commonly used criterion is to space the filters uniformally along the logarithmic frequency scale which is often justified from a human auditory perception point of view

An alternative criterion for designing a non-uniform filter bank is to use the critical band scale directly The spacing of the filters along the critical band is based on the perceptual studies and is intended to choose bands that give equal contribution to speech articulation The general shape of the critical band scale is given in Figure 3 4 The scale is close to linear for frequencies below about $1000Hz$ (i e , the bandwidth is essentially a constant as a function of f), and is close to logarithmic for frequencies above $1000Hz$ (i e , the bandwidth is essentially exponential as a function of f)

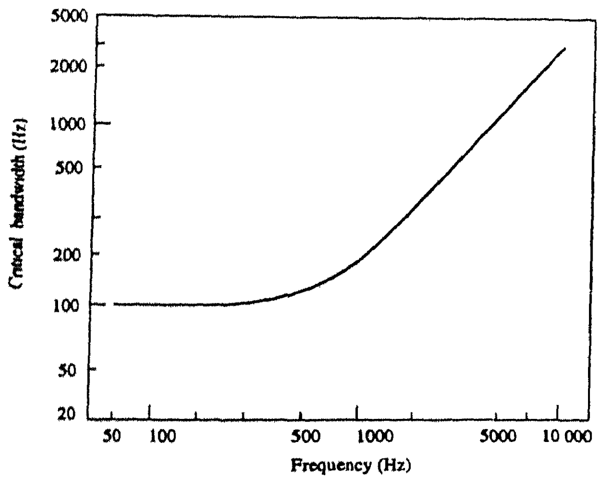


Figure 3 4: Critical band scale

A sight variation of this critical band scale, the *MEL-scale*, the most widely used and studied filter bank methods For this work we have also used MEL-scale based filter bank feature. Figure 3 5 shows a block diagram of this front end feature processor The basic steps required in this feature processing are as follows

1 *Preemphasis, Frame Blocking, and Windowing* These first three steps

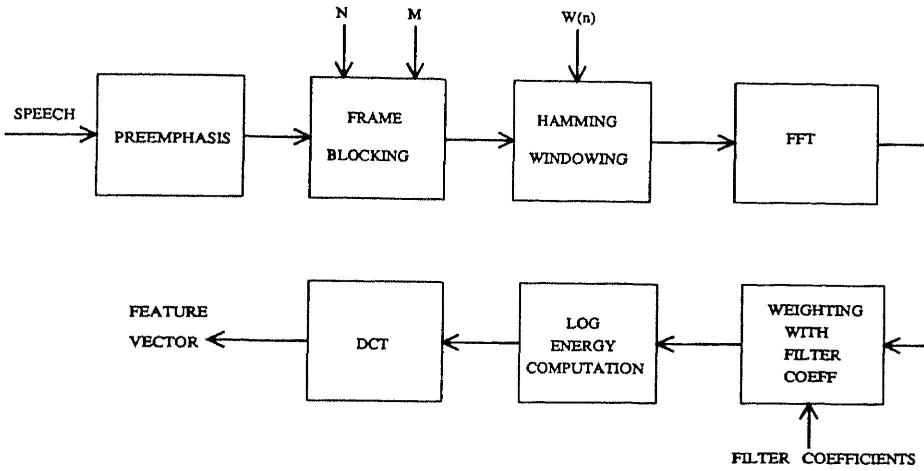


Figure 3 5. Block diagram of Mel-scale filter bank processor for speech recognition

are essentially the same as in the LPC processor and have been explained in detail in the LPC section earlier

2 *Calculation of Energy Vector* Each of the windowed waveform segments is transformed into the frequency domain by computing the FFT of the corresponding waveform A vector of log energies is then computed from each waveform segment by weighting the FFT coefficients by the magnitude frequency response of the filter bank The log energies are taken for the purpose of dynamic range compression and also in order to make the statistics of the estimated speech power spectrum approximately Gaussian

3 *Computing DCT*. The final processing stage is to apply the discrete cosine transform (DCT) to the log energy coefficients This has the effect of compressing the spectral information into the lower-order coefficients, and it also decorrelates them to allow the subsequent statistical modeling to use diagonal covariance matrices

Chapter 4

Database and Implementation

4.1 Database

For this task we have used "30k Numbers corpus" obtained from Oregon Graduate Institute (OGI) The numbers corpus is a collection of ordinal and cardinal numbers, continuous digit strings and isolated digit strings The utterances were taken from numerous telephone speech data collection efforts done by Center for Spoken Language Understanding (CSLU) at OGI This corpus contains 15,000 files. Each file has an orthographic transcription; about 7000 have a phonetic transcription The speech samples in database are sampled at 8 kHz

In our study, only the utterances which contain digits are used and the collection of these utterances will now be called as digit corpus The digit corpus is divided into three sets, namely training, development and test sets. In our study, training set consists of 3/5th of digit corpus while the share of development and test sets are 1/5th each The development set is further divided into four sub-sets, namely Dev, Dev1, Dev2, and Dev3 The set Dev contains all files that have phonetic labels. The other three sub-sets are obtained by splitting up all available developments files into three parts such that these resulting sub-sets are speaker- independent. In this study we have mainly used training set for model training and Dev1 for testing

4.2 Implementation

We have made use of "CSLU Speech Toolkit" which apart from many other utilities for speech processing also provides the development environment for the hidden Markov modeling based speech recognizers. This toolkit may be downloaded from http //www cse ogi edu/CSLU/toolkit

Defining the task

The grammar used by our continuous digit recognizer is depicted in Figure 4 1 This grammar can be used to recognize any number of spoken digits, with optional silence between words

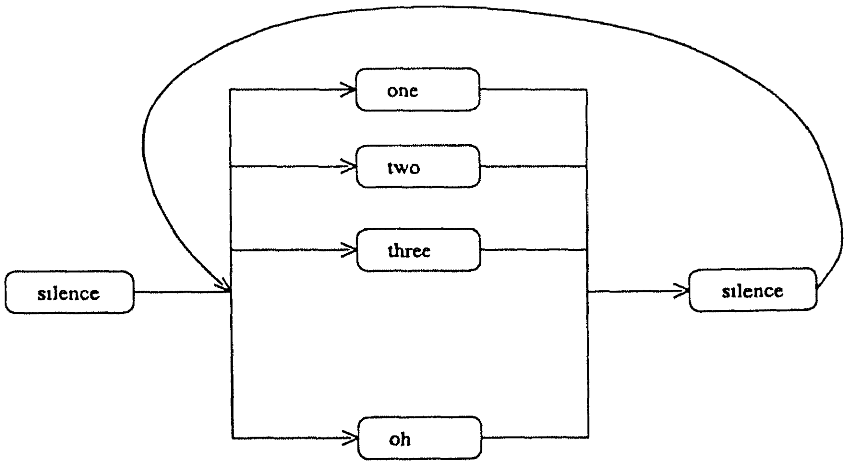


Figure 4 1 Continuous digit string grammar

Model Prototyping

The word pronunciation models for the words defined by the grammar are given in Table 4 1 These are defined using the ARPA phonetic alphabet Based on the pronunciation models we define the first set of HMM models required for the task

one	w ah n	seven	s eh v ah n
two	t uw	eight	ey t
three	th r iy	nine	n ay n
four	f aor	zero	z ih r ow
five	f ay v	oh	ow
six	s ih k s	sil	sil

Table 4 1 Word Pronunciations for the Digit Recognizer

Feature extraction

We have used two types of features in this work, namely LPC-cepstral coefficients and MEL-cepstral coefficients The different parameters settings used LPC- and MEL-cepstral feature extraction processes are given in Table 4 2 and Table 4 3 respectively In both cases along with base features, the first and second order time derivatives were also computed The time derivatives were computed over 5 frames Thus the size of feature vector for each frame is 36 in case of LPC and 39 for MEL We have also done cepstral mean subtraction in both cases

Window size	16 ms
Frame size	10 ms
Data preemphasis	0 98
Analysis order	10
No of cepstral coeff	12
Cepstral liftering coeff	0 6

Table 4 2 LPC-Cepstral Feature Generation Parameter Setting

Window size	16 ms
Frame size	10 ms
Data preemphasis	0 98
No of filters	21
No of cepstral coeff	13
Cepstral liftering coeff	0 6

Table 4 3 MEL-Cepstral Feature Generation Parameter Setting

Model Training

The first step to train the chosen mono phone models is to initialize them The initialization is a three step process First, all data (segments) for the particular class are loaded into the memory Each segment is then cut into equal sized segments, depending upon the number of states in the particular model All data allocated to a particular HMM state is then combined and the initial mixture mean vectors were estimated using vector quantization The initial mixture variances were set to the pool variance of the data During this step mixture weights and state transition probability matrices were not computed. After that the parameter estimates were further improved by Viterbi state alignment Finally, the model parameters were estimated using expectation/maximization (EM) algorithm

Allophonic variations

The seed models developed ın previous step do not dıstınguısh same sounds whıch are produced ın dıfferent parts of a word For ınstance, the "t" ın two wıll be pronounced dıfferently from the "t" ın eıght In thıs case the burst of sound "t" ın eıght ıs sometımes unreleased, and therefore needs to be modeled dıfferently Thıs can be done by clonıng the model "t", but ınstead of usıng standard left-to-rıght model, thıs thırd state ıs made optıonal

For the dıgıts whıch end ın nasal (one/seven/nıne) the speaker often tend to emphasıze the word's final phoneme These varıatıons are captured by clonıng the $<ah>$ model Table 4 4 lısts the new set of pronuncıatıons gıven the desıgn consıderatıons dıscussed

one	w ah n [ah3]	seven	s eh v ah2 n [ah3]
two	t uw	eıght	ey td
three	th r ıy	nıne	n ay n
four	f aor	zero	z ıh r ow
five	f ay v	oh	ow
sıx	s ıh ks	sıl	sıl

Table 4 4 Word Pronuncıatıons for the Dıgıt Recognızer based on Allophonıc Varıatıons

Transcription

The phonetıcally hand labeled data are typıcally only suffıcıent to create "seed" models for a phoneme-based recognızer To build a more accurate and more robust recognızer requıres much more data Most databases contaın word level transcrıptıon, which may be used to augment the exıstıng traınıng data For that a finıte state grammar is created usıng input word transcriptıon where each node or state ın the grammar contaıns a word and its pronunciation variants obtained by a pronunciation lexicon The standard Viterbi algorithm is then used to find the best possible path through the grammar which results in the selection of the pronunciation variants to fit the sentence

The resulting output word transcription may then be used to generate the associated model transcription for HMM embedded training

Embedded model re-estimation

Model initialization and single model training only use the data associated with the particular model During the training step it is assumed that the phonetic boundaries are defined and there is no interaction between neighboring models This problem is addressed in embedded parameter estimation step by creating a composite model by combining the models defined by the word transcription file for each of the training utterances specified Training then continues similar to the single model training Using the composite model, however, the results in all models updated simultaneously

Evaluation

The previous training steps creates a series of HMM models Now the model, which gives best performance when evaluated on previously unseen data, is selected To setup the evaluation process we first need to construct a task grammar and associated search network Then a Viterbi decoder searches through the finite state grammar to give recognition answers which are compared with input transcriptions to perform the scoring All extraneous speech labels are also suppressed during the scoring

The outline of the typical process of training an HMM model for speech recognition is shown in Figure 4 2.

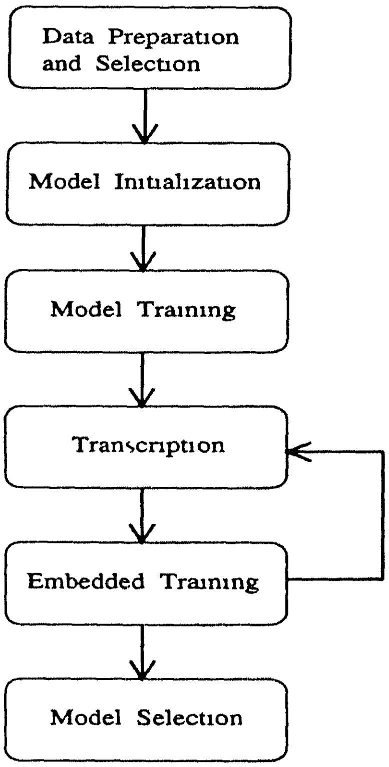


Figure 4 2 Outline of typical HMM training process

Chapter 5

Experimental Results

5.1 Results

We have implemented a continuous digit HMM recognizer and its performance is evaluated for LPC and MEL front-end features with different model size (number of states) and number of mixtures per state The performance (accuracies) of the recognition system for both LPC and MEL features are given in Table 5 1 and Table 5 2 respectively In these tables two accuracies are given corresponding to a particular number of state and a particular number of mixture per state, the top one is word accuracy and the bottom one is sentence accuracy The plots of word recognition accuracy as a function of the number of mixtures per state in the model for models with number of states 1, 3, 4, and 5 are given in Figure 5 1, Figure 5 2, Figure 5 3, and Figure 5 4 respectively The plot of corresponding sentence recognition accuracies is given in Figure 5 5

The salient inferences derived from this study are as follows

- The recognition accuracy with MEL-cepstral feature is better than with LPC-cepstral feature This improvement is in between 0 5-1% with simple phoneme/word grammar

- In case of digit recognition a word accuracy below 94% is far from satisfactory, therefore, our study indicates that a product of the number of states and the number of mixtures per state in a model below ten is inadequate for satisfactory performance

- The single mixture (multivariate normal density) is inadequate to represent speech parameter satisfactorily, this is in accordance with the results reported by Rabiner *et al* [20]

- Comparing the word accuracy with LPC- and MEL-cepstral features particularly at lower mixtures per state, one may notice that performance of LPC feature is better than that of MEL feature These results provide some indication that speech spectrum with LPC feature is less multi-modal than with MEL feature and also that MEL feature is more descriptive than LPC feature that is why it supersedes the LPC one at higher mixtures per state

- Comparing the sentence accuracy we found that the performance with MEL feature is also better than LPC feature on an average by 1%

- We also noticed that for a fixed product of the number of states and the number of mixtures per state in a model, models having higher number of states provide better word and as well as sentence accuracy than those containing less number of states This improvement is more pronounced when the product of the number of state and number of mixture per state in a model is below twelve

5.2 Conclusion

Our study indicates that MEL-cepstral feature performs better than LPC-cepstral feature though this improvement is not very significant with a model of sufficiently large size (number of states and number of mixtures per state) This study also indicates that a model size (4 states and 3 mixture per state) with MEL-cepstral feature is good enough for continuous digit recognition task as ıt provıdes suffıcıently hıgh accuracy (95%) wıth moderate model complexıty

No. of	**No. of Mıxtures per State**				
States	**1**	**2**	**3**	**4**	**5**
1	53 03% 18 22%	75 07% 33 10%		83 07% 44 50%	
2					91 17% 71 07%
3	87 98% 58 41%	90 0% 72 18%		93 86% 78 02%	
4			94 29% 79 69%		94 99% 82 75%
5	91 14% 67 59%	94 03% 78 44%		95 13% 82 06%	

Table 5 1 Recognition Accuracıes (Word & Sentence) wıth LPC Feature for Models wıth Dıfferent States and Mıxtures per State

No. of	No. of Mixtures per State				
States	1	2	3	4	5
1	49 35% 13 35%	74 61% 28 79%		85 19% 48 96%	
2					92 17% 71 20%
3	86 44% 53 41%	92 98% 73 71%		94 82% 80 67%	
4			95 54% 83 30%		95 49% 84 01%
5	90 41% 65 23%	94 36% 79 28%		95 59% 83 45%	

Table 5 2 Recognition Accuracies (Word & Sentence) with MEL Feature for Models with Different States and Mixtures per State

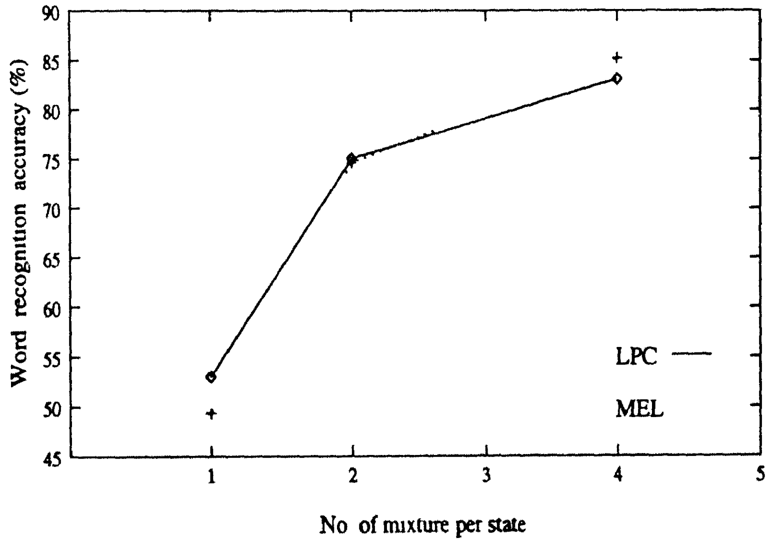


Figure 5 1 Word recognition accuracy versus the number of mixture per state for model with 1 state

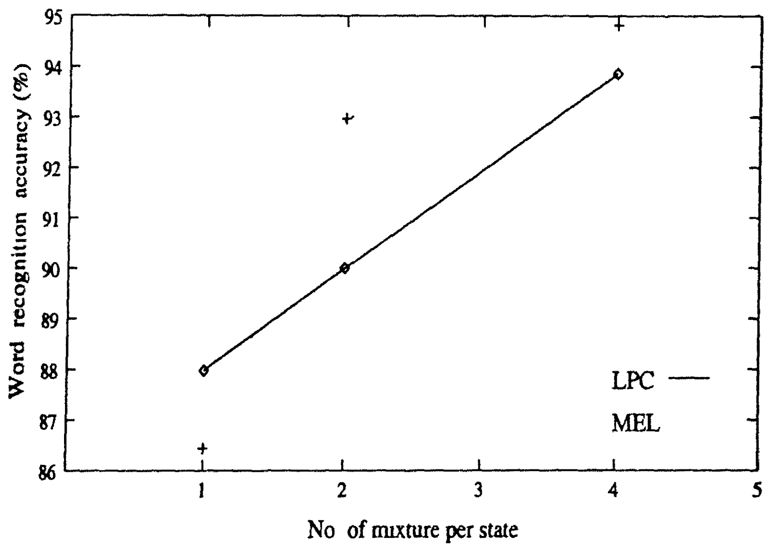


Figure 5 2 Word recognition accuracy versus the number of mixture per state for model with 3 states

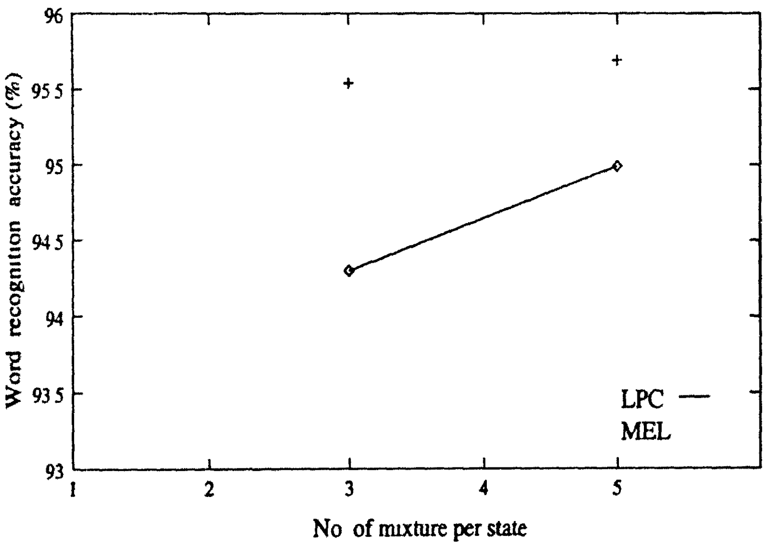


Figure 5 3 Word recognition accuracy versus the number of mixture per state for model with 4 states

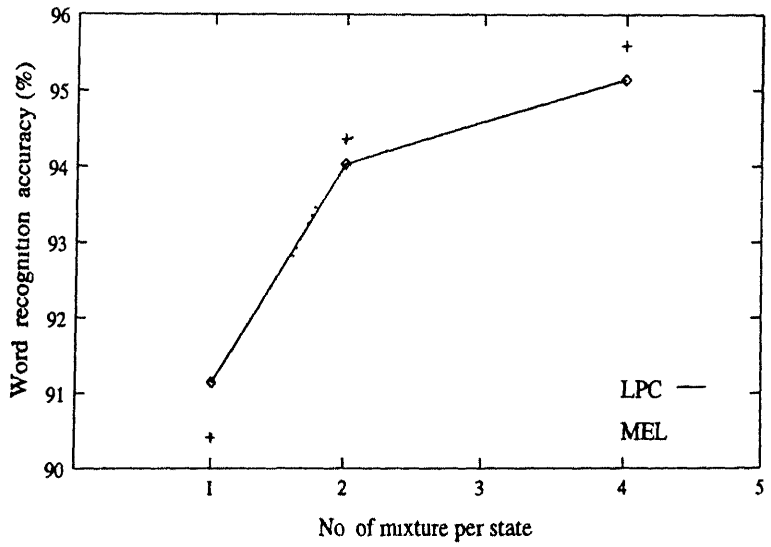


Figure 5 4 Word recognition accuracy versus the number of mixture per state for model with 5 states

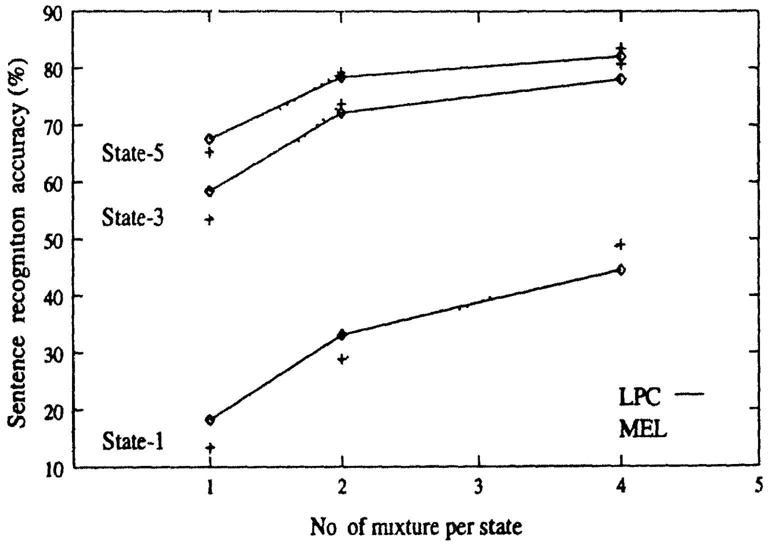


Figure 5 5. Sentence recognition accuracy versus the number of mixture per state for models with 1, 3, and 5 states

Chapter 6

Study of Relationship between Speakers

6.1 Motivation

Since last five decade a lot of research has already been done to solve the problem of speech recognition and a large amount of understanding has been developed in this process, but the complete solution of the problem still remains elusive There are still some areas in which the understanding till date is insufficient such as human auditory mechanism and speech production process

One of the major problems in speech recognition is variability of speech Though human beings are able to handle the variability in speech very well, machines have not been able to overcome this problem fully Thus in order to understand and parameterize the variability, one may get the motivation to ask, given any two speaker what is the relationship between them in terms of speech related parameters This query not only befits the human nature of enquiry, but is also of great importance in the field of speech recognition by machine, particularly in the speaker independent speech recognition systems

It is well known that different speakers have different formant frequencies for the same vowel [18] The vowels are almost completely characterized by the first three formant (resonance) frequencies of the vocal tract transfer function These formant frequencies are in turn affected by the length of the pharyngeal-oral tract, the narrowness of the tract, and the constriction along the tract Out of these factors, the difference in vocal-tract length for different speakers is attributed to be the main source of variability among different speakers The reason behind the above attribution is the gross approximation of the vocal-tract of a speaker by a uniform tube of characteristic length, so that the resonant frequencies in the speech produced by a speaker are inversely proportional to the length of his/her vocal-tract [26] Experimental observations have reported the length of vocal-tact typically 18 cm in males and 13 cm in females [25]

The above idea is reported in the literature as "scale relationship" among speakers and is demonstrated as a valid model of the speech spectrum [1, 25] Many researchers used the scale model of the speech spectrum and reported improvement in recognition accuracy by estimating the scale-factor and normalizing the spectra of different speakers [4, 10, 14] This is, however, an expensive procedure since the scale factor has to be estimated for every speaker with respect to reference speaker Recently Umesh *et al* [23] have suggested the use of features based on scale transform that do not require the explicit computation of scale factor and avoids the computation complexity The basic idea used in all such systems is to derive features which help provide insensitivity to the scale factor existing among the speech spectra of different speakers Such scale invariant features are obtained through the use of some suitable transforms viz, Scale-transform, Mellin-transform [3]

Even with this much understanding about the relationship among different speakers, the problem of speaker variability is still not completely solved A recent study about scale relationship has reported the frequency dependence of the scale factor relating different speakers in scale relationship [22] Thus scale model of the speech spectrum is an approximation of the ideal model and further research is necessary to identify this ideal model

As a first step towards this goal, we have tried to see that if there is some simple relationship between spectra of any two speakers In trying to 'dentify this simple relationship, we have looked at the large class of simple models and have not been biased by the physiological considerations or otherwise

6.2 Database

We have used a database obtained by Gordon Peterson and Harold Barney [18] It contains the values of first three formant frequencies for 10 vowels /IY/, /IH/, /EH/, AE/, /AH/, /AA/, /AO/, /UH/, /UW/, and /ER/, derived from the speech collected from 33 male, 28 female and 15 child speakers For our study, we have divided the above database into three subsets, namely DB1, DB2, and DB3 Subset DB1 consists of 8 male speakers, 8 female speakers and 8 child speakers while subset DB2 also contains same number of male, female, and child speakers but these are different from DB1 except one child speaker which common to both sets as there were only 15 child speakers in the database Subset DB3 is the same as DB1 except for all speakers only first two formant frequencies are considered

6.3 Method of Search

Since we had no idea about the functional form and the number of parameters in the probable relationship among speakers at the outset We even didn't know there at all exists a unique relationship among speakers or not Thus we had extremely big task in hand Therefore to limit the task, we had decided to search for only simple relationship and that too not containing more than four parameters Some justification for such limited search can be derived from the two facts First, a relationship with higher parameter count is more likely to fit the experimental data than the one with lower parameter count and secondly it would be extremely difficult to provide some physical explanation to a fancy relationship

To search for the best fit simple relationships to our speech data we have used a curve fitting software package named "TableCurve 2D v4 0" In our search we have tried to fit the simple curves only to the data which is the collection of the values of the formant frequencies for 10 vowels There were two set of values for all formants per speaker Thus we have formed a vector of 60 elements (10 vowels $\times$ 3 formants $\times$ 2 examples) in case of datasets DB1 and DB2 while a vector of 40 elements in case of dataset DB3 as only first two formants were considered

The procedure followed the search is summarized as follows

- In every dataset, we first took first four male speakers and first four female speakers, then every male speaker was associated once with all the four female speakers and the lists of fit equations were noted in each case

- The coefficient of least square error was used as the criterion for finding the fit simple equations in each case

- In all cases only those resulting fit equations were considered in the study, for which the coefficient of least square error was not below three percent of the error coefficient of the best fit equation

- Then we took the remaining four male speakers and first four child speakers and associated them to in the same manner as explained previously and the fit equations were noted as per above laid criteria

- After that the remaining four female speakers were associated with the remaining four child speakers were associated and fit equations were noted similarly as above

- In this way we had noted 48 different lists of fit simple equations We had also reversed the association in each of the above cases Therefore we finally had 96 different lists of fit equations with each datasets

Following the above procedures we are left with 20-30 fit equations in each case. Now to narrow down our search further we have followed certain thumb rules to attach score to any particular fit equation

Firstly, if the coefficient of least square error of any equation in short-list of fit equations is equal up to two decimal places to that of the best fit equation in that short-list , a score of 1 is awarded and if its error coefficient is greater than by one at the second place from decimal than that of best fit one, a score of 2 is given and so on

Secondly, all those fit equations, which are equal up to two decimal places, are further segregated by adding 0 5 to their score due to first rule depending on whether their error coefficient is above the half of the range or not

In this way we found the associated score with every equation in the short-listed fit equations in each case and then for every dataset few best fit equation are listed based on their total accumulated score The lower the score the better fit that equation for a particular dataset

6.4 Results

We have searched for the fit simple equations in the manner explained in the previous section The list of the top 15 fit equations for all the three datasets are given in Table 6 1 In this table the fit equations are expressed in terms of x and y parameters which represent the formant frequencies of any two chosen speakers

Following inferences can be deduced from our study

- This study indicates that the equation ($y = ax^b$) figures out to be the best relationship among speakers

- The scale relationship ($y = ax$), which is well accepted and possesses physical explanation, is the special case of the equation($y = ax^b$)

- This study also validates the scale-relationship among speakers as it appeared among the top ten relationships in all datasets

- The equation ($y = a + bx$) and those resembling its form have shown better performance than those highly different in form This provides

Fit Equation	Relative Order of Fit		
Formula	DB1	DB2	DB3
$y = ax^b$	I	I	I
$y = a + b\exp^{-x/c}$	II	I	I
$y = a + bx/\ln x$	II	II	I
$y^{0.5} = a + bx^{0.5}$	II	II	II
$y = a + bx$	III	II	I
$\ln y = a + b/\ln x$	III	III	IV
$y = a + bx\ln x$	III	III	II
$\ln y = a + b\ln x$	IV	III	III
$y^{0.5} = a + b(\ln x)^2$	IV	IV	III
$y^{0.5} = a + bx^{0.5}\ln x$	IV	IV	II
$y = ax$	IV	IV	III
$y = a + bx^c$	V	V	I
$y^2 = a + bx^2$	V	V	II
$y^2 = a + bx^2\ln x$	V	V	IV
$y^{-1} = a + b/x$	VI	V	IV

Table 6.1: List of fit equations for datasets DB1, DB2, and DB3

some clue that the ultimate relation may be close in form to the linear transformation.

- Comparing the performance of different fit equations across all the three datasets one may easily notice that all most all listed equations have relatively consistence except equation ($y = a+bx^c$) which has suddenly jumped to order I in case of dataset DB3. The one possible reason that we could able to point out, is the variability of the third formant value in the database. The first two formant values exhibit significantly less variability and lie mostly along the curve exhibited by equation ($y = a + bx^c$).

6.5 Conclusion

In this study we have tried to narrow down the choice of the probable relationship based on formant frequencies between any two speakers. Such a study might provide clues in finding appropiate transforms to remove speaker dependencies and improve speaker independent speech recognition. A future study using the whole formant envelopes might throw more light and in turn provide more reliable relationships.

Bibliography

[1] P. Bamberg. Vocal tract normalization. Technical report, Verbex, 1981.

[2] L. E. Baum. An inequality and associated maximization technique in statistical estimation of probabilistic functions of markov processes. *Inequalities*, pages 1–8, 1972.

[3] Jingdong Chen, Bo Xu, and Taiyi Huang. A novel robust feature of speech signal based on the mellin transform for speaker-independent speech recognition. In *Proc. ICASSP'98*, Washington, USA, 1998.

[4] Ellen Eide and Herbert Gish. A Parametric Approach to Vocal Tract Length Normalization. In *Proc. IEEE ICASSP'96*, pages 346–349, Atlanta, USA, May 1996.

[5] G. D. Forney. The viterbi algorithm. *Proc. IEEE*, 61:268–278, March 1973.

[6] S. Furui. Speaker independent isolated word recognition using dynamic features of speech spectrum. *IEEE Trans. Acoust., Speech, Signal Processing*, 34(1):52–59, Feb. 1986.

[7] X. D. Huang, Y. Ariki, and M. A. Jack. *Hidden Markov Models For Speech Recognition*. Edinburgh University Press, 1990.

[8] B. H. Juang and L. R. Rabiner. The segmental k-means algorithm for estimating parameters of hidden markov models. *IEEE Trans. Acoust., Speech, Signal Processing*, 38(9):1639–1641, Sept. 1990.

[9] B. H. Juang, L. R. Rabiner, and J. G. Wilpon. On the use of bandpass liftering in speech recognition. *IEEE Trans. Acoust., Speech, Signal Processing*, 35(7):947–954, July 1987.

[10] T. Kamm, G. Andreou, and J. Cohen. Vocal Tract Normalization in Speech Recognition: Compensating for Systematic Speaker Variability. In *Proc. of the 15th Annual Speech Research Symposium*, pages 175–178, Johns Hopkins University, Baltimore, June 1995.

[11] D. Klatt. Review of the arpa speech understanding project. *J. Acoustic Soc. America*, 62:1345–1366, 1977.

[12] C. H. Lee and L. R. Rabiner. A network-based frame-synchronous level building algorithm for connected word recognition. In *Proc. ICASSP 88*, pages 410–413, April 1988.

[13] Kai-Fu Lee. *Automatic Speech Recognition: The Development of the SPHINX System.* Kluwer Academic Publishers, Boston, 1989.

[14] Li Lee and Richard C. Rose. Speaker Normalization Using Efficient Frequency Warping Procedures. In *Proc. IEEE ICASSP'96*, pages 353–356, Atlanta, USA, May 1996.

[15] S. E. Levinson, L R. Rabiner, and M. M. Sondhi. An introduction to the application of the theory of probabilistic function of a markov process to automatic speech recognition. *Bell System Tech. J.*, 62(4):1035–1074, April 1983.

[16] J. D. Markel and A. H. Gray. *Linear Prediction of Speech.* Springer-Verlag, 1976.

[17] D. B. Paul. The lincon robust continuous speech recognizer. In *Proc. ICASSP-89*, pages 449–452, Glasgow, Scotland, 1989.

[18] G. E. Peterson and H. L. Barney. Control methods used in a study of the vowels. *J. Acoust. Soc. America*, 24(2):175–194, March 1952.

[19] L. Rabiner and B. H. Juang. *Fundamentals Of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[20] L. R. Rabiner, B. H. Juang, S. E. Levinson, and M. M. Sondhi. Recognition of isolated digits using hidden markov models with continuous mixture densities. *AT&T Technical Journal*, 64(6):1211–1233, July-August 1985.

[21] L. R. Rabiner, J. G. Wilpon. and F. K. Soong. High performance connected digit recognition using hidden markov models. In *Proc. ICASSP 88*, April 1988.

[22] S. Umesh, L. Cohen, N. Marinovic, and D. Nelson. Frequency-Warping in Speech. In *Proc. International Conference on Spoken Language Processing*, Philadelphia, USA, 1996.

[23] S. Umesh, L. Cohen, N. Marinovic, and D. Nelson. Scale Transform In Speech Analysis. *IEEE Transactions on Speech and Audio Processing*, January 1999.

[24] A. J. Viterbi. Error bounds for conventional codes and asymptotically optimal decoding algorithm. *IEEE Trans. Information Theory*, IT-13:260–269, April 1967.

[25] H. Wakita. Normalization of vowels by vocal-tract length and its application to vowel identification. *IEEE Trans. Acoustic, Speech, Signal Processing*, ASSP-25(2):183–192, April 1977.

[26] S. Wegmann, D. McAllaster, J. Orloff, and B. Peskin. Speaker normalization on conversational telephone speech. In *ICASSP-96*, volume 1, pages 339–341, 1996.

[27] V. W. Zue. The use of speech knowledge in automatic speech recognition. *Proc. IEEE*, 73:1602–1615, 1985.